An Introduction to (Hybrid) Probabilistic Logic Programming

Luc De Raedt

IJCAI - Hybrid Reasoning 2015
Overview

• Part I: An introduction to Prob. Logic Programming and the relation to alternative frameworks

• Part II: Inference

• Part III: Learning

• Part IV: Dynamics & Continuous distributions for Relational Tracking (in Robotics)

• Focus on ProbLog line of research at KU Leuven
PART 1: Intro to PLP
Biomine Network

Notch receptor processing

Biological Process

GO:0007220 - participates_in

0.220

presenilin 2

Gene

EntrezGene:81751

Gene

EntrezGene:81751
Phenetic

- Causes: Mutations
- All related to similar phenotype
- Effects: Differentially expressed genes
- 27,000 cause effect pairs

- Interaction network:
  - 3063 nodes
  - Genes
  - Proteins
  - 16,794 edges
  - Molecular interactions
  - Uncertain

- Goal: connect causes to effects through common subnetwork
  - = Find mechanism
  - Techniques:
    - DTProbLog
    - Approximate inference

Can we find the mechanism connecting causes to effects?

[De Maeyer et al., Molecular Biosystems 13, NAR 15]
Example: Information Extraction

<table>
<thead>
<tr>
<th>instance</th>
<th>iteration</th>
<th>date learned</th>
<th>confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>kelly_andrews is a female</td>
<td>826</td>
<td>29-mar-2014</td>
<td>98.7</td>
</tr>
<tr>
<td>investment_next_year is an economic sector</td>
<td>829</td>
<td>10-apr-2014</td>
<td>95.3</td>
</tr>
<tr>
<td>shibenik is a geopolitical entity that is an organization</td>
<td>829</td>
<td>10-apr-2014</td>
<td>97.2</td>
</tr>
<tr>
<td>quality_web_design_work is a character trait</td>
<td>826</td>
<td>29-mar-2014</td>
<td>91.0</td>
</tr>
<tr>
<td>mercedes_benz_cls_by_carlsson is an automobile manufacturer</td>
<td>829</td>
<td>10-apr-2014</td>
<td>95.2</td>
</tr>
<tr>
<td>social_work is an academic program at the university rutgers_university</td>
<td>827</td>
<td>02-apr-2014</td>
<td>93.8</td>
</tr>
<tr>
<td>dante wrote the book the_divine_comedy</td>
<td>826</td>
<td>29-mar-2014</td>
<td>93.8</td>
</tr>
<tr>
<td>willie_aames was born in the city los_angeles</td>
<td>831</td>
<td>16-apr-2014</td>
<td>100.0</td>
</tr>
<tr>
<td>kitt_peak is a mountain in the state or province arizona</td>
<td>831</td>
<td>16-apr-2014</td>
<td>96.9</td>
</tr>
<tr>
<td>greenwich is a park in the city london</td>
<td>831</td>
<td>16-apr-2014</td>
<td>100.0</td>
</tr>
</tbody>
</table>

instances for many different relations

degree of certainty

NELL: http://rtw.ml.cmu.edu/rtw/
Graphs & Randomness

ProbLog, Phenetic, Prism, ICL, Probabilistic Databases, ...

- all based on a “random graph” model

Stochastic Logic Programs, ProPPR, PCFGs, ...

- based on a “random walk” model
- connected to PageRank
- not the subject of this talk!
Probabilistic Logic Programming

Distribution Semantics [Sato, ICLP 95]: probabilistic choices + logic program → distribution over possible worlds

e.g., PRISM, ICL, ProbLog, LPADs, CP-logic, ...

- multi-valued switches
- probabilistic alternatives
- probabilistic facts
- annotated disjunctions
- causal-probabilistic laws
ProbLog by example:

A bit of gambling

- toss (biased) coin & draw ball from each urn
- win if (heads and a red ball) or (two balls of same color)

Probabilistic fact: heads is true with probability 0.4 (and false with 0.6)
Probabilistic fact: first ball is red with probability 0.3 and blue with 0.7

Annotated disjunction: second ball is red with probability 0.2, green with 0.3, and blue with 0.5

Logical rule encoding consequences

---

0.4 :: heads.
0.3 :: col(1,red); 0.7 :: col(1,blue) <- true.
0.2 :: col(2,red); 0.3 :: col(2,green);
  0.5 :: col(2,blue) <- true.

win :- heads, col(_,red).
win :- col(1,C), col(2,C).
Questions

0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue) <- true.
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue) <- true.

win :- heads, col(_,red).
win :- col(1,C), col(2,C).

marginal probability

• Probability of win

conditional probability

• Probability of win given col(2,green)?

• Most probable world where win is true?

MPE inference
Possible Worlds

0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue) <- true.
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue) <- true.

win :- heads, col(_,red).
win :- col(1,C), col(2,C).

$0.4 \times 0.3 \times 0.3$
Possible Worlds

0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue) <- true.
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue) <- true.

win :- heads, col(_,red).
win :- col(1,C), col(2,C).

\[ 0.4 \times 0.3 \times 0.3 \quad (1-0.4)\times 0.3 \times 0.2 \quad (1-0.4)\times 0.3 \times 0.3 \]
All Possible Worlds

0.024
H  R  R
W

0.036
R  R
W

0.056
H  B  R
W

0.084
B  R

0.036
H  R  G
W

0.054
R  G

0.084
H  B  G

0.126
B  G

0.060
H  R  B
W

0.090
R  B

0.140
H  B  B
W

0.210
B  B
\[ P(\text{win}) = \sum = 0.562 \]
\[ P(\text{win} | \text{col(2, green)}) = \frac{?}{\Sigma} \]
\[ = P(\text{win} \land \text{col(2, green)}) / P(\text{col(2, green)}) \]

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.024</td>
<td>0.036</td>
<td>0.056</td>
<td>0.084</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H R R W</td>
<td>R R W</td>
<td>H B R W</td>
<td>B R</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>0.036</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H R G W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>0.060</strong></td>
<td>0.090</td>
<td>0.140</td>
<td>0.210</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H R B W</td>
<td>R B W</td>
<td>H B B W</td>
<td>B B W</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Most likely world where \textit{win} is true?

\begin{align*}
\begin{array}{cccc}
0.024 & 0.036 & 0.056 & 0.084 \\
\text{H} & \text{R} & \text{R} & \text{W} \\
\text{R} & \text{R} & \text{W} \\
\text{H} & \text{B} & \text{R} & \text{W} \\
\text{B} & \text{R} & \text{W} \\
0.036 & 0.054 & 0.084 & 0.126 \\
\text{H} & \text{R} & \text{G} & \text{W} \\
\text{R} & \text{G} & \text{W} \\
\text{H} & \text{B} & \text{G} & \text{W} \\
\text{B} & \text{G} & \text{W} \\
0.060 & 0.090 & 0.140 & 0.210 \\
\text{H} & \text{R} & \text{B} & \text{W} \\
\text{R} & \text{B} & \text{W} \\
\text{H} & \text{B} & \text{B} & \text{W} \\
\text{B} & \text{B} & \text{W} \\
\end{array}
\end{align*}
Most likely world where col(2, blue) is false?

MPE Inference

0.024

\[
\begin{array}{c}
H \\
R \\
R \\
W
\end{array}
\]

0.036

\[
\begin{array}{c}
H \\
R \\
R \\
W
\end{array}
\]

0.056

\[
\begin{array}{c}
H \\
B \\
R \\
W
\end{array}
\]

0.084

\[
\begin{array}{c}
H \\
B \\
R \\
W
\end{array}
\]

0.036

\[
\begin{array}{c}
H \\
R \\
G \\
W
\end{array}
\]

0.054

\[
\begin{array}{c}
H \\
R \\
G \\
W
\end{array}
\]

0.084

\[
\begin{array}{c}
H \\
B \\
G \\
W
\end{array}
\]

0.126

\[
\begin{array}{c}
H \\
B \\
G \\
W
\end{array}
\]

0.060

\[
\begin{array}{c}
H \\
R \\
B \\
W
\end{array}
\]

0.090

\[
\begin{array}{c}
H \\
R \\
B \\
W
\end{array}
\]

0.140

\[
\begin{array}{c}
H \\
B \\
B \\
W
\end{array}
\]

0.210

\[
\begin{array}{c}
H \\
B \\
B \\
W
\end{array}
\]
**Distribution Semantics**

(with probabilistic facts)

\[ P(Q) = \sum_{F \cup R \models Q} \prod_{f \in F} p(f) \prod_{f \notin F} (1 - p(f)) \]

query

sum over possible worlds where \(Q\) is true

subset of probabilistic facts

Prolog rules

probability of possible world

[Sato, ICLP 95]
cProbLog: constraints on possible worlds

weight(skis,6).
weight(boots,4).
weight(helmet,3).
weight(gloves,2).

P::pack(Item) :-
    weight(Item,Weight),
    P is 1.0/Weight.

excess(Limit) :- ...

not excess(10).
pack(helmet) v pack(boots).

cProbLog: constraints as FOL formulas treat as evidence

[Fierens et al, PP 12; Shterionov et al]
Alternative view: CP-Logic

\[
\begin{align*}
\text{throws(john).} & \quad 0.5::\text{throws(mary).} \\
0.8 :: \text{break} & \leftarrow \text{throws(mary).} \\
0.6 :: \text{break} & \leftarrow \text{throws(john).}
\end{align*}
\]

\[
P(\text{break}) = 0.6 \times 0.5 \times 0.8 + 0.6 \times 0.5 \times 0.2 + 0.6 \times 0.5 + 0.4 \times 0.5 \times 0.8
\]
E.g., “throwing a rock at a glass breaks it with probability 0.3 and misses it with probability 0.7”

\[(\text{Broken}(G):0.3) \lor (\text{Miss} 0.7) \leftarrow \text{ThrowAt}(G).\]

Note that the actual non-deterministic event ("rock flying at glass") is implicit.
Semantics

\[ (\text{Broken}(G) \ 0.3) \lor (\text{Miss} \ 0.7) \leftarrow \text{ThrowAt}(G) \]

Probability tree is an execution model of theory iff:
- Each tree-transition matches causal law
- The tree cannot be extended
- Each execution model defines the same probability distribution over final states

Slides CP-logic courtesy Joost Vennekens
ProbLog by example:

Rain or sun?

\[
\begin{align*}
0.5::\text{weather}(\text{sun}, 0) & ; 0.5::\text{weather}(\text{rain}, 0). \\
0.6::\text{weather}(\text{sun}, T) & ; 0.4::\text{weather}(\text{rain}, T) \\
& \quad : - T > 0, \ T\text{prev is } T-1, \ \text{weather}(\text{sun}, T\text{prev}). \\
0.2::\text{weather}(\text{sun}, T) & ; 0.8::\text{weather}(\text{rain}, T) \\
& \quad : - T > 0, \ T\text{prev is } T-1, \ \text{weather}(\text{rain}, T\text{prev}).
\end{align*}
\]

infinite possible worlds! BUT: finitely many partial worlds suffice to answer any given ground query
Possible worlds

\[ P = P_1 + P_2 + P_3 + P_4 \]

\[ P_1 = 0.12 \]

\[ P_2 = 0.16 \]

\[ P_3 = 0.04 \]

\[ P_4 = 0.32 \]

?- weather(rain,2).

\[ P = P_1 + P_2 + P_3 + P_4 \]
Distributional Clauses (DC)

- Discrete- and continuous-valued random variables

\[
\text{length(Obj) } \sim \text{ gaussian}(6.0, 0.45) :\text{ type(Obj, glass).}
\]

\[
\text{stackable(Obj1, Obj2) } :\text{ length(Obj1) } \geq \text{ length(Obj2)},
\text{ width(Obj1) } \geq \text{ width(Obj2}).
\]

\[
\text{ontype(Obj, plate) } \sim \text{ finite}([0 : \text{ glass}, 0.0024 : \text{ cup}, 0 : \text{ pitcher}, 0.8676 : \text{ plate}, 0.0284 : \text{ bowl}, 0 : \text{ serving}, 0.1016 : \text{ none}]),
\text{ obj(Obj), on(Obj, O2), type(O2, plate).}
\]

random variable with

discrete distribution

defines a generative process (as for CP-logic)
Tree can become infinitely wide
Sampling
Well-defined under reasonable comparing values of random variables

\[\text{[Gutmann et al, TPLP 11; Nitti et al, IROS 13]}\]
Distributional Clauses (DC)

• Defines a generative process (as for CP-logic)
• Tree can become infinitely wide
  • Sampling …
• Well-defined under reasonable assumptions
ProbLog

- probabilistic choices + their consequences
- probability distribution over possible worlds
- how to efficiently answer questions?
  - most probable world (MPE inference)
  - probability of query (computing marginals)
  - probability of query given evidence
Summary: ProbLog Syntax

- input database: ground facts
  \[\text{person(bob)}.\]

- probabilistic facts
  \[0.5::\text{stress(bob)}.\]

- annotated disjunctions
  \[0.5::\text{stress}(X) :- \text{person}(X).
  0.4::a(X); 0.3::b(X); 0.2::c(X); 0.1::d(X) :- q(X).
  0.5::\text{weather(sun,0)} ; 0.5::\text{weather(rain,0)}.\]

- flexible probabilities
  \[P::\text{pack(Item)} :- \text{weight(Item,W)}, P \text{ is } 1.0/W.\]

- Prolog clauses
  \[\text{smokes}(X) :- \text{influences}(Y,X), \text{smokes}(Y).
  \text{excess}([I|R],\text{Limit}) :- \text{\#+pack(I)}, \text{excess}(R,\text{Limit}).\]
Probabilistic Logic Programming

Distribution Semantics [Sato, ICLP 95]: probabilistic choices + logic program → distribution over possible worlds

OVERVIEW paper [Kimmig, De Raedt, Arxiv, MLJ 15]

e.g., PRISM, ICL, ProbLog, LPADs, CP-logic, ...

- multi-valued switches
- probabilistic alternatives
- probabilistic facts
- annotated disjunctions
- causal-probabilistic laws
Probabilistic databases

programming versus database query language
different types of queries

<table>
<thead>
<tr>
<th>Company</th>
<th>Product</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>sony</td>
<td>walkman</td>
<td>0.96</td>
</tr>
<tr>
<td>microsoft</td>
<td>mac_os_x</td>
<td>0.96</td>
</tr>
<tr>
<td>ibm</td>
<td>personal_computer</td>
<td>0.96</td>
</tr>
<tr>
<td>microsoft</td>
<td>mac_os</td>
<td>0.9</td>
</tr>
<tr>
<td>adobe</td>
<td>adobe_inDesign</td>
<td>0.9</td>
</tr>
<tr>
<td>adobe</td>
<td>adobe_dreamweaver</td>
<td>0.87</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Company</th>
<th>City</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>microsoft</td>
<td>redmond</td>
<td>1.00</td>
</tr>
<tr>
<td>ibm</td>
<td>san_jose</td>
<td>0.99</td>
</tr>
<tr>
<td>emirates_airlines</td>
<td>dubai</td>
<td>0.93</td>
</tr>
<tr>
<td>honda</td>
<td>torrance</td>
<td>0.93</td>
</tr>
<tr>
<td>horizon</td>
<td>seattle</td>
<td>0.93</td>
</tr>
<tr>
<td>egyptair</td>
<td>cairo</td>
<td>0.93</td>
</tr>
<tr>
<td>adobe</td>
<td>san_jose</td>
<td>0.93</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Example from Suciu et al 2011

```
select x.Product, x.Company
from ProducesProduct x, HeadquarteredIn y
where x.Company=y.Company and y.City='san_jose'
```
Probabilistic Programs

• Distributional clauses / PLP similar in spirit
  • to e.g. BLOG, ... but embedded in existing logic and programming language
  • to e.g. Church but use of logic instead of functional programming ...

• natural possible world semantics and link with prob. databases.

• somewhat harder to do meta-programming
Church by example:

**A bit of gambling**

- Toss a biased coin and draw a ball from each urn
- Win if (heads and a red ball) or (two balls of the same color)

```
(define heads (mem (lambda () (flip 0.4))))
(define color1 (mem (lambda () (if (flip 0.3) 'red 'blue))))
(define color2 (mem (lambda ()
                        (multinomial '(red green blue) '(0.2 0.3 0.5))))))
(define redball (or (equal? (color1) 'red) (equal? (color2) 'red)))
(define win1 (and (heads) redball))
(define win2 (equal? (color1) (color2)))
(define win (or win1 win2))
```
Markov Logic

Key differences

• programming language

• Pro(b)log uses least-fix point semantics
  • can express transitive closure of relation
  • this cannot be expressed in FOL (and Markov Logic), requires second order logic

• \( p(X,Y) :- p(X,Z), p(Z,Y). \)
PART II: Inference
Inference in PLP

- As in Prolog and logic programming
  - proof-based
- As in Answer Set Programming
  - model based
- As in Probabilistic Programming
  - sampling
Given:
- program
- queries
- evidence

Find:
- marginal probabilities
- conditional probabilities
- MPE state

1. using proofs
2. using models

Inference

logical reasoning
data structure
probabilistic inference

knowledge
compilation
Proofs in ProbLog

\[ 0.8 : : \text{stress(ann)}. \]
\[ 0.6 : : \text{influences(ann,bob)}. \]
\[ 0.2 : : \text{influences(bob,carl)}. \]

\[
\text{smokes}(X) :- \text{stress}(X).
\text{smokes}(X) :- \text{influences}(Y,X), \text{smokes}(Y).
\]

?- \text{smokes(carl)}.  
?- \text{stress(carl)}.
?- \text{influences}(Y,\text{carl}), \text{smokes}(Y).

\[
\text{influences}(\text{bob,carl}) \& \text{influences}(\text{ann,bob}) \& \text{stress(ann)}
\]

probability of proof = \(0.2 \times 0.6 \times 0.8 = 0.096\)
Proofs in ProbLog

\[ \text{proofs overlap! cannot sum probabilities (disjoint-sum-problem)} \]
Disjoint-Sum-Problem

possible worlds

solution: knowledge compilation

\[ \text{sum of proof probabilities: } 0.096 + 0.08 = 0.1760 \]
Binary Decision Diagrams  [Bryant 86]

• compact graphical representation of Boolean formula

• automatically disjoins proofs

• popular in many branches of CS
Current Approach
(ProbLog2)

Find relevant ground program for queries & evidence

Weighted CNF

use weighted model counting / satisfiability

0.4::heads(1).
0.7::heads(2).
0.5::heads(3).
win :- heads(1).
win :- heads(2), heads(3).

win :- heads(1).
win :- heads(2), heads(3).

\[
\text{win} \leftrightarrow \text{h}(1) \lor (\text{h}(2) \land \text{h}(3))
\]

may require loop-breaking

\[
(\neg \text{win} \lor \text{h}(1) \lor \text{h}(2)) \\
\land (\neg \text{win} \lor \text{h}(1) \lor \text{h}(3)) \\
\land (\text{win} \lor \neg \text{h}(1)) \\
\land (\text{win} \lor \neg \text{h}(2) \lor \neg \text{h}(3))
\]

use standard tool

h(1) \rightarrow 0.4 \quad h(2) \rightarrow 0.7 \quad h(3) \rightarrow 0.5
\neg h(1) \rightarrow 0.6 \quad \neg h(2) \rightarrow 0.3 \quad \neg h(3) \rightarrow 0.5

[Fierens et al, TPLP 14]
ProbLog $\rightarrow$ CNF

0.8::stress(ann).
0.4::stress(bob).
0.6::influences(ann,bob).
0.2::influences(bob,carl).

?- smokes(carl).

smokes(X) :- stress(X).
smokes(X) :- influences(Y,X), smokes(Y).

• Find relevant ground rules by backward reasoning

smokes(carl) :- influences(bob,carl), smokes(bob).
smokes(bob) :- stress(bob).
smokes(bob) :- influences(ann,bob), smokes(ann).
smokes(ann) :- stress(ann).

• Convert to propositional logic formula

sm(c) ↔ (i(b,c) ∧ sm(b))
∧ sm(b) ↔ (st(b) ∨ (i(a,b) ∧ sm(a)))
∧ sm(a) ↔ st(a)

• Rewrite in CNF (as usual)
Weighted Model Counting

Given a propositional formula \( \phi \) in conjunctive normal form (CNF) and a ProbLog program & query, the probability of the query \( Q \) is given by:

\[
P(Q) = \sum_{F \cup R = Q} \prod_{f \in F} p(f) \prod_{f \notin F} (1 - p(f))
\]

And the weighted model count (WMC) is:

\[
WMC(\phi) = \sum_{I_V \models \phi} \prod_{l \in I_V} w(l)
\]

where interpretations (truth value assignments) of propositional variables define possible worlds.

For a literal \( p::f \), the weight is:

- \( w(f) = p \)
- \( w(\neg f) = 1 - p \)
WMC using d-DNNFs

1. represent formula as d-DNNF
2. transform into arithmetic circuit
3. evaluate bottom-up

\[ \text{alarm} \leftrightarrow \text{burglary} \lor \text{earthquake} \]
\[ \text{calls}(\text{john}) \leftrightarrow \text{alarm}, \text{hears\_alarm}(\text{john}) \]
\[ \text{calls}(\text{john}) \]

[Figure: Fierens et al, TPLP 14]
Current Approach
(ProbLog2)

Find relevant ground program for queries & evidence

Weighted CNF

use weighted model counting / satisfiability

0.4::heads(1).
0.7::heads(2).
0.5::heads(3).

win :- heads(1).
win :- heads(2), heads(3).

win :- heads(1).
win :- heads(2), heads(3).

\[
\text{win} \leftrightarrow h(1) \lor (h(2) \land h(3))
\]

may require loop-breaking

\[
\begin{align*}
\neg \text{win} \lor h(1) \lor h(2) \\
\land \neg \text{win} \lor h(1) \lor h(3) \\
\land \text{win} \lor \neg h(1) \\
\land \text{win} \lor \neg h(2) \lor \neg h(3)
\end{align*}
\]

use standard tool

h(1) \rightarrow 0.4 \quad h(2) \rightarrow 0.7 \quad h(3) \rightarrow 0.5

\neg h(1) \rightarrow 0.6 \quad \neg h(2) \rightarrow 0.3 \quad \neg h(3) \rightarrow 0.5

[Fierens et al, TPLP 14]
Inference for
DC

\begin{align*}
n &\sim \text{uniform}([1, 2, 3, 4, 5, 6, 7, 8, 9, 10]). \\
\text{color}(X) &\sim \text{uniform}([\text{grey, blue, black}]) \leftarrow \text{material}(X) \sim= \text{metal}. \\
\text{color}(X) &\sim \text{uniform}([\text{black, brown}]) \leftarrow \text{material}(X) \sim= \text{wood}. \\
\text{material}(X) &\sim \text{finite}([0.3:\text{wood}, 0.7:\text{metal}]) \leftarrow n \sim= N, \text{between}(1, N, X). \\
\text{drawn}(Y) &\sim \text{uniform}(L) \leftarrow n \sim= N, \text{findall}(X, \text{between}(1, N, X), L). \\
\text{size}(X) &\sim \text{beta}(2, 3) \leftarrow \text{material}(X) \sim= \text{metal}. \\
\text{size}(X) &\sim \text{beta}(4, 2) \leftarrow \text{material}(X) \sim= \text{wood}. \\
\end{align*}
Sampling

- $P(\text{query})$?

$$P(\text{query}) \approx \frac{\text{# query holds}}{\text{# worlds sampled}}$$
Rejection Sampling

• $P(\text{query} \mid \text{evidence})$?
Rejection Sampling

- \( P(\text{query} \mid \text{evidence}) \) ?

  evidence holds

  evidence does not hold
Rejection Sampling

\[ P(\text{query} \mid \text{evidence}) \approx \frac{\# \text{ query & evidence holds}}{\# \text{ evidence holds}} \]
Likelihood Weighting

- $P(\text{query} \mid \text{evidence})$?
Likelihood Weighting

• $P(\text{query} \mid \text{evidence})$ ?
LW for DC

Given a goal $G$ and the global variables $w_q^{(i)}, iq, x^{P(i)}$, applying a rule produces a new goal $G'$ and modifies the global variables:

1. $G'$ is the new goal obtained from $G$ using a kind of SLD-resolution step;
2. if a new variable $r$ is sampled with value $v$,
   - set $w_q^{(i)} \leftarrow w_q^{(i)} \frac{p(r=v|x^{P(i)})}{g(r=v|x^{P(i)})}$ (based on LW) and
   - $x^{P(i)} \leftarrow x^{P(i)} \cup \{r = v\}$. 
   In addition, if $r = Val \in iq$ with $r$ grounded, then:
   - $iq \leftarrow iq\theta$ with $\theta = \{Val = v\}$.
3. if a new atom $a$ is proved set $x^{P(i)} \leftarrow x^{P(i)} \cup \{a\}$. 
can cope with evidence like color(1) = color(2)
and size(1) = 0.356, size(1)=size(2), …
outperforms BLOG … unification + LW
1: (color(2) \sim= black); w_q^{(i)} = 1; x^{P(i)} = \emptyset

2b on (7):
2: (material(2) \sim= metal, sample(color(2), D_{color(2)}), color(2) \sim= black); w_q^{(i)} = 1; x^{P(i)} = \emptyset

2b on (9):
3: (n \sim= N, between(1, N, 2), sample(material(2), D_{material(2)}), material(2) \sim= metal, sample(color(2), D_{color(2)}), color(2) \sim= black); w_q^{(i)} = 1; x^{P(i)} = \emptyset

2b on (6):
4: (sample(n, D_{n}), n \sim= N, between(1, \sim(n), 2), sample(material(2), D_{material(2)}), material(2) \sim= metal, sample(color(2), D_{color(2)}), color(2) \sim= black); w_q^{(i)} = 1; x^{P(i)} = \emptyset

3b:
5: (n \sim= 3, between(1, 3, 2), sample(material(2), D_{material(2)}), material(2) \sim= metal, sample(color(2), D_{color(2)}), color(2) \sim= black); w_q^{(i)} = 1; x^{P(i)} = \{n = 3\}

2a followed by 1a
6: (sample(material(2), D_{material(2)}), material(2) \sim= metal, sample(color(2), color(2) \sim= black); w_q^{(i)} = 1; x^{P(i)} = \{n = 3\}

3b:
7: (material(2) \sim= metal, sample(color(2), D_{color(2)}), color(2) \sim= black)

w_q^{(i)} = 1; x^{P(i)} = \{n = 3, material(2) = wood\}

fail, backtracking to 1
Introduction.
Probabilistic logic programs are logic programs in which some of the facts are annotated with probabilities.

ProbLog is a tool that allows you to intuitively build programs that do not only encode complex interactions between a large sets of heterogenous components but also the inherent uncertainties that are present in real-life situations.

The engine tackles several tasks such as computing the marginals given evidence and learning from (partial) interpretations. ProbLog is a suite of efficient algorithms for various inference tasks. It is based on a conversion of the program and the queries and evidence to a weighted Boolean formula. This allows us to reduce the inference tasks to well-studied tasks such as weighted model counting, which can be solved using state-of-the-art methods known from the graphical model and knowledge compilation literature.

The Language. Probabilistic Logic Programming.
ProbLog makes it easy to express complex, probabilistic models.

0.3::stress(X) :- person(X).
0.2::influences(X,Y) :- person(X), person(Y).
PART III: Learning
Parameter Learning

e.g., webpage classification model

for each \text{CLASS1}, \text{CLASS2} and each \text{WORD}

\begin{align*}
\text{link\_class}(\text{Source}, \text{Target}, \text{CLASS1}, \text{CLASS2}). \\
\text{word\_class}(\text{WORD}, \text{CLASS}).
\end{align*}

\text{class}(\text{Page}, \text{C}) \; : \; \text{has\_word}(\text{Page}, \text{W}), \text{word\_class}(\text{W}, \text{C}).

\text{class}(\text{Page}, \text{C}) \; : \; \text{links\_to}(\text{OtherPage}, \text{Page}), \text{class}(\text{OtherPage}, \text{OtherClass}), \text{link\_class}(\text{OtherPage}, \text{Page}, \text{OtherClass}, \text{C}).
Sampling
Interpretations
Parameter Estimation

\[ p(\text{fact}) = \frac{\text{count}(\text{fact is true})}{\text{Number of interpretations}} \]
Learning from partial interpretations

- Not all facts observed
- Soft-EM
- use expected count instead of count
- $P(Q|E)$ -- conditional queries!

[Gutmann et al, ECML 11; Fierens et al, TPLP 14]
Learning from single facts / entailment

- Only true facts are given; e.g. as in HMM
- key setting in PRISM, also in ProbLog
- EM-based, variations exist
- use expected count instead of count
- \( P(Q \mid E) \) -- conditional queries!

[Gutmann et al, ECML 08; Sato 95]
Bayesian Parameter Learning

- Learning as inference (e.g., Church)
- Prior distributions for parameters
- Given data, find most likely parameter values
Information Extraction in NELL

### Recently-Learned Facts

<table>
<thead>
<tr>
<th>Instance</th>
<th>Iteration</th>
<th>Date Learned</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>kelly_andrews is a female</td>
<td>826</td>
<td>29-mar-2014</td>
<td>98.7</td>
</tr>
<tr>
<td>investment_next_year is an economic sector</td>
<td>829</td>
<td>10-apr-2014</td>
<td>95.3</td>
</tr>
<tr>
<td>shibenik is a geopolitical entity that is an organization</td>
<td>829</td>
<td>10-apr-2014</td>
<td>97.2</td>
</tr>
<tr>
<td>quality_web_design_work is a character trait</td>
<td>826</td>
<td>29-mar-2014</td>
<td>91.0</td>
</tr>
<tr>
<td>mercedes_benz_cl_s_by_carlsson is an automobile manufacturer</td>
<td>829</td>
<td>10-apr-2014</td>
<td>95.2</td>
</tr>
<tr>
<td>social_work is an academic program at the university rutgers_university</td>
<td>827</td>
<td>02-apr-2014</td>
<td>93.8</td>
</tr>
<tr>
<td>dante wrote the book the_divine_comedy</td>
<td>826</td>
<td>29-mar-2014</td>
<td>93.8</td>
</tr>
<tr>
<td>willie_aames was born in the city los_angeles</td>
<td>831</td>
<td>16-apr-2014</td>
<td>100.0</td>
</tr>
<tr>
<td>kitt_peak is a mountain in the state or province arizona</td>
<td>831</td>
<td>16-apr-2014</td>
<td>96.9</td>
</tr>
<tr>
<td>greenwich is a park in the city london</td>
<td>831</td>
<td>16-apr-2014</td>
<td>100.0</td>
</tr>
</tbody>
</table>

instances for many different relations

degree of certainty

NELL: http://rtw.ml.cmu.edu/rtw/
Rule learning in NELL

• Original approach
  • Make probabilistic data deterministic
  • run classic rule-learner (variant of FOIL)
  • re-introduce probabilities on learned rules and predict
Probabilistic Rule Learning

• Learn the rules directly in a PLP setting

• Generalize relational learning and inductive logic programming directly towards probabilistic setting

• Traditional rule learning/ILP as a special case

• Apply to probabilistic databases like NELL

• Approach in PPR (Cohen et al) and in ProbLog (IJCAI-15)
In order to test probabilistic rule learning for facts extracted by NELL, we used the NELL athlete dataset, which has already been used in the context of meta-interpretive learning of higher-order dyadic Datalog \cite{36}. This dataset contains 10,130 facts. The number of facts per predicate is listed in Table 5. The unary predicates in this dataset are deterministic, whereas the binary predicates have a probability attached.

**Table 5: Number of facts per predicate (NELL athlete dataset)**

<table>
<thead>
<tr>
<th>Predicate</th>
<th>Facts</th>
</tr>
</thead>
<tbody>
<tr>
<td>athletecoach(person, person)</td>
<td>18</td>
</tr>
<tr>
<td>athleteplaysport(person, sport)</td>
<td>1921</td>
</tr>
<tr>
<td>athleteplaysinleague(person, league)</td>
<td>872</td>
</tr>
<tr>
<td>coachesinleague(person, league)</td>
<td>93</td>
</tr>
<tr>
<td>teamhomestadium(team, stadium)</td>
<td>198</td>
</tr>
<tr>
<td>athleteplayssportsteamposition(person, position)</td>
<td>255</td>
</tr>
<tr>
<td>athlete(person)</td>
<td>1909</td>
</tr>
<tr>
<td>coach(person)</td>
<td>624</td>
</tr>
<tr>
<td>male(person)</td>
<td>7</td>
</tr>
<tr>
<td>organization(league)</td>
<td>1</td>
</tr>
<tr>
<td>personafria(person)</td>
<td>1</td>
</tr>
<tr>
<td>personaustralia(person)</td>
<td>22</td>
</tr>
<tr>
<td>personeurope(person)</td>
<td>1</td>
</tr>
<tr>
<td>personus(person)</td>
<td>6</td>
</tr>
<tr>
<td>sportsleague(league)</td>
<td>18</td>
</tr>
<tr>
<td>sportsteamposition(position)</td>
<td>22</td>
</tr>
<tr>
<td>athleteplaysforteam(person, team)</td>
<td>721</td>
</tr>
<tr>
<td>teamplaysinleague(team, league)</td>
<td>1085</td>
</tr>
<tr>
<td>athletealsoknownas(person, name)</td>
<td>17</td>
</tr>
<tr>
<td>coachesteam(person, team)</td>
<td>132</td>
</tr>
<tr>
<td>teamplayssport(team, sport)</td>
<td>359</td>
</tr>
<tr>
<td>athleteplaysinleague(person, league)</td>
<td>872</td>
</tr>
<tr>
<td>athletehomestadium(person, stadium)</td>
<td>187</td>
</tr>
<tr>
<td>attraction(stadium)</td>
<td>2</td>
</tr>
<tr>
<td>female(person)</td>
<td>2</td>
</tr>
<tr>
<td>hobby(sport)</td>
<td>5</td>
</tr>
<tr>
<td>person(person)</td>
<td>2</td>
</tr>
<tr>
<td>personasia(person)</td>
<td>4</td>
</tr>
<tr>
<td>personcanada(person)</td>
<td>1</td>
</tr>
<tr>
<td>personmexico(person)</td>
<td>108</td>
</tr>
<tr>
<td>sport(sport)</td>
<td>36</td>
</tr>
<tr>
<td>sportsteam(team)</td>
<td>1330</td>
</tr>
<tr>
<td>stadiumoreventvenue(stadium)</td>
<td>171</td>
</tr>
</tbody>
</table>

Table 5 also shows the types that were used for the variables in the base declarations for the predicates. As indicated in Section 4.5, this typing of the variables forms a syntactic restriction on the possible groundings and ensures that arguments are only instantiated with variables of the appropriate type. Furthermore, the \texttt{LearnRule} function of the ProbFOIL algorithm is based on \texttt{mFOIL} and allows to incorporate a number of variable constraints. To reduce the search space, we imposed that unary predicates that are added to a candidate rule during the learning process can only use variables that have already been introduced. Binary predicates can introduce at most one new variable.

### 6.2 Relational probabilistic rule learning

In order to illustrate relational probabilistic rule learning with ProbFOIL in the context of NELL, we will learn rules and report their respective accuracy for each binary predicate with more than 500 facts. In order to show ProbFOIL's speed, also the runtimes are reported. Unless indicated otherwise, both the \texttt{m-estimate}'s \texttt{m} value and the beam width were set to 1. The value of \texttt{p} for rule significance was set to 0.9. The rules are postprocessed such that only range-restricted rules are obtained. Furthermore, to avoid a bias towards the majority class, the examples are balanced, i.e., negative examples are added to balance the number of positives.
Fig. 5: Histogram of probabilities for each of the binary predicates with more than 500 facts: (a) athleteplaysforteam; (b) athleteplayssport; (c) teamplaysinleague; and, (d) athleteplaysinleague.

6.2.1 athleteplaysforteam(person,team)

0.875:: athleteplaysforteam(A,B) :- coachesteam(A,B).
0.99080:: athleteplaysforteam(A,B) :- teamhomestadium(B,C), athletehomestadium(A,C).
0.75:: athleteplaysforteam(A,B) :- teamhomestadium(B,C), male(A), athleteplayssport(A,B).
0.75:: athleteplaysforteam(A,B) :- teamhomestadium(B,C), athleteplaysinleague(A,C), teamplaysinleague(B,C), athlete(A).
0.75:: athleteplaysforteam(A,B) :- teamplayssport(B,C), athleteplayssport(A,C), coach(A), teamplaysinleague(B,C).
0.97555:: athleteplaysforteam(A,B) :- personus(A), teamplayssport(B,C).
0.762:: athleteplaysforteam(A,B) :- teamplayssport(B,C), athleteplayssport(A,C), personmexico(A), teamplaysinleague(B,C).
0.52571:: athleteplaysforteam(A,B) :- teamplayssport(B,C), athleteplayssport(A,C), athleteplaysinleague(A,C), teamplaysinleague(B,C), athlete(A).
0.50546:: athleteplaysforteam(A,B) :- teamplayssport(B,C), teamplaysinleague(B,C), athlete(A), teamplayssport(B,C).
0.50:: athleteplaysforteam(A,B) :- teamplayssport(B,C), teamplaysinleague(B,C), athlete(A), teamplayssport(B,C).
0.50:: athleteplaysforteam(A,B) :- teamplayssport(B,C), teamhomestadium(B,C), athleteplayssport(A,C), teamplaysinleague(B,C).
0.52941:: athleteplaysforteam(A,B) :- teamplayssport(B,C), teamhomestadium(B,C), coach(A), teamplaysinleague(B,C).
0.55287:: athleteplaysforteam(A,B) :- teamplayssport(B,C), teamplaysinleague(B,C), athleteplaysinleague(A,C), athlete(A).
0.46875:: athleteplaysforteam(A,B) :- teamplayssport(B,C), teamplaysinleague(B,C), coach(A), teamhomestadium(B,C).
Part IV: Dynamics
Dynamics: Evolving Networks

- **Travian**: A massively multiplayer real-time strategy game
  - Commercial game run by TravianGames GmbH
  - ~3,000,000 players spread over different “worlds”
  - ~25,000 players in one world

[Thon et al., MLJ 11, ECML 08]
World Dynamics

Fragment of world with

~10 alliances
~200 players
~600 cities

alliances color-coded

Can we build a model of this world?
Can we use it for playing better?

[Thon, Landwehr, De Raedt, ECML08]
World Dynamics

Fragment of world with

~10 alliances
~200 players
~600 cities

alliances color-coded

Can we build a model of this world?
Can we use it for playing better?

[Thon, Landwehr, De Raedt, ECML08]
World Dynamics

Fragment of world with

~10 alliances
~200 players
~600 cities

alliances color-coded

Can we build a model of this world?
Can we use it for playing better?

[Thon, Landwehr, De Raedt, ECML08]
Causal Probabilistic Time-Logic (CPT-L)

how does the world change over time?

one of the effects holds at time $T+1$

0.4::conquest(Attacker,C); 0.6::nil <-
\[
\text{city}(C,\text{Owner}), \text{city}(C2,\text{Attacker}), \text{close}(C,C2).
\]

if cause holds at time $T$

[Thon et al, MLJ 11]
Social Network of Chats
Limitations CPT-L

Inference slow / scalability

- uses knowledge compilation method
- compile formula for \( P(I_{t+1} | I_{[0,t]}) \)
- exponential in number of time steps

No continuous distributions

- needed for robotics / relational tracking applications
Relational Tracking

- Track people or objects over time? Even if temporarily hidden?
- Recognize activities?
- Infer object properties?
Relational State Estimation over Time

**Box scenario**
- object tracking even when invisible
- estimate spatial relations

**Magnetism scenario**
- object tracking
- category estimation from interactions

[Nitti et al, IROS 13]
Speed 0x

Queries
(updated every 5 steps)

Box ID=4
Cube ID=3
Magnetic scenario

- 3 object types: magnetic, ferromagnetic, nonmagnetic
- Nonmagnetic objects do not interact
- A magnet and a ferromagnetic object attract each other
- Magnetic force that depends on the distance
- If an object is held magnetic force is compensated.
DC Particle Filter (DCPF)

Dynamic Distributional Clauses

Particle Filter

DCPF

Flexible (relational) state representation

Fast inference (state estimation) in general models

Goal

“A particle filter for hybrid relational domains” IROS 2013
D. Nitti, T. De Laet, L. De Raedt
Dynamic Distributional Clauses

Prior distribution \( p(x_0) \)
State transition model \( p(x_t|x_{t-1}, u_t) \)
Measurement model \( p(z_t|x_t) \)
Other rules: \( p(x'_t|x''_t) \)
Magnetic scenario

- 3 object types: magnetic, ferromagnetic, nonmagnetic

\[ \text{type}(X)_t \sim \text{finite}([1/3:\text{magnet}, 1/3:\text{ferromagnetic}, 1/3:\text{nonmagnetic}]) \leftarrow \text{object}(X). \]

- 2 magnets attract or repulse

\[ \text{interaction}(A,B)_t \sim \text{finite}([0.5:\text{attraction}, 0.5:\text{repulsion}]) \leftarrow \text{object}(A), \text{object}(B), A<B, \text{type}(A)_t = \text{magnet}, \text{type}(B)_t = \text{magnet}. \]

- Next position after attraction

\[
\begin{align*}
\text{pos}(A)_{t+1} & \sim \text{gaussian}(&\text{middlepoint}(A,B)_t, \text{Cov}) \leftarrow \\
& \text{near}(A,B)_t, \text{not}(\text{held}(A)), \text{not}(\text{held}(B)), \\
& \text{interaction}(A,B)_t = \text{attr}, \\
& c/dist(A,B)_t^2 > \text{friction}(A)_t.
\end{align*}
\]

\[
\text{pos}(A)_{t+1} \sim \text{gaussian}(\text{pos}(A)_t, \text{Cov}) \leftarrow \text{not}(\text{attraction}(A,B)).
\]
Particle Filter
(Sequential Monte Carlo)

- Based on sampling $\rightarrow$ approximate inference
- Particles (samples) to represent $\text{bel}(x_t)$

Particle $i \sim \text{bel}(x_{t-1})$ $\xrightarrow{\text{Sampling}}$ $x^{(i)}_{t-1}$ $\xrightarrow{u_t}$ $x^{(i)}_t$ $\xrightarrow{\text{Weighting}}$ $p(z_t | x_t)$ $\xrightarrow{z_t}$ Resampling
Classical Particle Filter vs DCPF

- **Classical PF**
  - Fixed set of random variables
  - Update the entire state

- **DCPF**
  - *Adaptive state (particle):* the number of facts / random variables can change over time
  - Particles are partial interpretations
  - Expressive language

\[
\begin{align*}
X_t^{(i)} & = \begin{pmatrix} 1.1 \\ 2.3 \\ 10.3 \end{pmatrix} \\
X_{t+1}^{(i)} & = \begin{pmatrix} 1.2 \\ 2.1 \\ 10.5 \end{pmatrix}
\end{align*}
\]
Optimized inference: partial state

Distributional Clauses Particle Filter (DCPF)

Sampled

Marginalized

Classical particle filter

Pos(1)=(0, 3)
Pos(2)=(0, 1)

right(X,Y)
near(X,Y)
interaction(X,Y)
type(X) ~ [1/3:magnet,...]

Pos(1)=(0, 2)
Pos(2)=(0, 1)
near(1,2)=true
type(1)=nonmagnetic

right(X,Y)
near(X,Y)
interaction(X,Y)
type(X) ~ [1/3:magnet,...]

Pos(1)=(0, 3)
Pos(2)=(0, 1)
near(1,2)=false
near(2,1)=false
interaction(1,2)=none
type(1)=nonmagnetic
type(2)=nonmagnetic

Pos(1)=(0, 2)
Pos(2)=(0, 1)
near(1,2)=true
near(2,1)=true
interaction(1,2)=none
interaction(2,1)=none
type(1)=nonmagnetic
type(2)=nonmagnetic
Inference in DCPF

Two steps:

Query $p(z_{t+1}|x_{t+1})$ (weighting + part of sampling step)
Query $p(x_{t+1}|x_t, u_{t+1})$ (to complete the sampling step)
using the DC inference
particles are partial interpretations

$\text{bel}(x_t)$ fully represented by $\{x_t^{(i)}\} \cup \text{Program}$

History $\{x_{0:t-1}^{(i)}\}$ not necessary

Issue: particles (interpretations) may grow till becoming complete
Experiments

- Particles are partial state, remaining variables are marginalized
- Better performance in bigger models
Ongoing Work

- Online parameter learning [Nitti, ICRA 2014]
- Integrate with planning [Nitti, ECML 15, EWRL 15]
- Applications in robotics (also to learn affordances)
Learning relational affordances

Learn probabilistic model

From two object interactions

Generalize to N

Moldovan et al. ICRA 12, 13, 14
Occluded Object Search

- Models of objects and their spatial arrangement
- different types of objects suitable for different tasks
- shelves with objects of different shape and size
- given a task, find an object to perform that task

[Moldovan et al ICRA 2014]
ProbLog for activity recognition from video

• Separation between low-level events (LLE) and high-level events (HLE)
  ➢ LLE: walking, running, active, inactive, abrupt
  ➢ HLE: meeting, moving, fighting, leaving_object
• Probabilistic Logic approach: Event Calculus in ProbLog (Prob-EC) to infer the high-level events from an algebra of low-level events.
• Example:

\[
\text{initiatedAt(} fighting(P_1, P_2) = \text{true}, T) \leftarrow \\
\text{happensAt(} abrupt(P_1), T), \\
\text{holdsAt(} close(P_1, P_2, 44) = \text{true}, T), \\
\text{not happensAt(} inactive(P_2), T).
\]

[Skarlatidis et al, TPLP 13]
Thanks!

http://dtai.cs.kuleuven.be/problog
• PRISM http://sato-www.cs.titech.ac.jp/prism/

• ProbLog2 http://dtai.cs.kuleuven.be/problog/

• Yap Prolog http://www.dcc.fc.up.pt/~vsc/Yap/ includes
  • ProbLog1

• cplint https://sites.google.com/a/unife.it/ml/cplint

• CLP(BN)

• LP2

• PITA in XSB Prolog http://xsb.sourceforge.net/

• AILog2 http://artint.info/code/ailog/ailog2.html

• SLPs http://stoics.org.uk/~nicos/sware/pepl

• contdist http://www.cs.sunysb.edu/~cram/contdist/

• DC https://code.google.com/p/distributional-clauses

• WFOMC http://dtai.cs.kuleuven.be/ml/systems/wfomc
References


